Text Clustering for Topic Detection

Young-Woo Seo Katia Sycara

CMU-RI-TR-04-03

January 2004

Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **JAN 2004**	2. REPORT TYPE	3. DATES COVERED **00-00-2004 to 00-00-2004**

4. TITLE AND SUBTITLE **Text Clustering for Topic Detection**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University,Robotics Institute,Pittsurgh,PA,15213**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S)
	11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The world wide web represents vast stores of information. However, the sheer amount of such information makes it practically impossible for any human user to be aware of much of it. Therefore, it would be very helpful to have a system that automatically discovers relevant, yet previously unknown information, and reports it to users in human-readable form. As the first attempt to accomplish such a goal, we proposed a new clustering algorithm and compared it with existing clustering algorithms. The proposed method is motivated by constructive and competitive learning from neural network research. In the construction phase, it tries to find the optimal number of clusters by adding a new cluster when the intrinsic difference between the instance presented and the existing clusters is detected. Each cluster then moves toward the optimal cluster center according to the learning rate by adjusting its weight vector. From the experimental results on the three different real world data sets, the proposed method shows an even trend of performance across the different domains, while the performance of our algorithm on text domains was better than that reported in previous research.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**18**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Abstract

The world wide web represents vast stores of information. However, the sheer amount of such information makes it practically impossible for any human user to be aware of much of it. Therefore, it would be very helpful to have a system that automatically discovers relevant, yet previously unknown information, and reports it to users in human-readable form. As the first attempt to accomplish such a goal, we proposed a new clustering algorithm and compared it with existing clustering algorithms. The proposed method is motivated by constructive and competitive learning from neural network research. In the construction phase, it tries to find the optimal number of clusters by adding a new cluster when the intrinsic difference between the instance presented and the existing clusters is detected. Each cluster then moves toward the optimal cluster center according to the learning rate by adjusting its weight vector. From the experimental results on the three different real world data sets, the proposed method shows an even trend of performance across the different domains, while the performance of our algorithm on text domains was better than that reported in previous research.

Contents

1 Introduction

With the rapid progress of computer technology in recent years, there has been an explosion of electronic information published on the Internet. The world wide web represents vast stores of information. However, the sheer amount of such information makes it practically impossible for any human user to be aware of much of it. Neither the depth nor the extent of possibly useful information is known to the human user of the web. This problem will only get worse, without efforts to deal with such information overload.

In particular, such a phenomenon is more serious for the decision maker, who has to determine what to do at the right time. Timely decision-making can be accomplished by awareness of the ever-changing surroundings. In order to make a decision on a timely basis, the decision maker or decision supporter should monitor information broadly and track, in depth, specific items relevant to the decision. However, considering the tremendous volume of surrounding information, it is practically impossible for decision makers or decision supporters to discover and monitor all of the pertinent information or knowledge. Therefore, it would be very helpful to have a system that automatically discovers relevant, yet previously unknown information, and reports it to users in human-readable form.

Topic Detection and Tracking (TDT) was proposed as the endeavor to find the solution for problem of "well-awareness" on a user's surroundings [1], [13]. The topic detection is the problem of identifying stories in several continuous news streams that pertain to new or previously unidentified events. It consists of discovering previously unidentified events in an accumulated collection ("retrospective detection"), or flagging the onset of new events from live news feeds ("on-line detection"). Both forms of detection by design lack advance knowledge of the new events, but have access to (unlabeled) data in chronological order. Given the facts that it should deal with unlabeled data and a subset of news reports with similar contents is grouped together, the clustering algorithms are a good choice to discover unknown events. For the purpose of our goal that finds out what happens in the past, the retrospective detection is the right direction to pursue.

In this paper, the approaches for event detection are presented as the first attempt at building a system for providing awareness of of the surrounding infosphere. This system should be able to group the incoming data into a cluster of items of similar content. Then, it should report the contents of the cluster in summarized human-readable form. Finally, it should be able to track events of interest in order to take advantage of developments. To accomplish this, we applied the newly devised and the existing clustering algorithms to the retrospective detection and compared them with each other within the standard metrics.

The rest of the paper is organized as follows. Section 2 describes how the text documents and the clusters are represented in computational form. Section 3 details the proposed clustering method and the existing ones. The experiments and conclusions are described by Section 4 and 5 respectively.

2 Data Representation

This section describes how a text document is represented in machine-readable form. The data representation for use by clustering algorithms that are discussed in this paper are based on the conventional (real-valued) vector space model [12]. This is one of the most widely used models for information retrieval because of its conceptual simplicity and the appeal of the underlying metaphor of using spatial proximity for semantic similarity.

In this model, each document is represented in a high-dimensional space, in which each dimension of the space corresponds to a word (a unigram) in the document set. This model is realized by means of the word-by-document matrix, $M = T \times N$, where there are T word features and N documents, $W = \{w_1, ..., w_t, ..., w_T\}$ and $D = \{\vec{d}_1, ..., \vec{d}_i, ..., \vec{d}_N\}, \vec{d}_i \in R^T$ respectively. The word feature set (W) is constructed by eliminating infrequent words and high frequency words. The elimination of words indicates that words are only considered as features, if they occur greater number of times than indicated by *frequent threshold* or at most smaller number of times than indicated by *infrequent threshold*. These threshold values are determined empirically at Section 4.

Each word has its weight w_t that indicates how important it is for a given text learning task. A variant of TFIDF (Term Frequency $\times$ Inverse Document Frequency) [12] is used for calculating a weight. The idea of this weighting method is to ensure that the weight of a word is scaled from 0.0 to 1.0 while preserving the original idea of TFIDF, which gives a word higher weight if it is frequently appeared in a document and less frequently occurred across the document collection. The weight of a word, w_t defined as:

$$w_t = \frac{(1 + \log(tf_{i,t})) \times \log \frac{N}{df_t}}{\sqrt{\sum_{s \neq t} (\log tf_{i,s} + 1)^2}} \tag{1}$$

where $tf_{i,t}$ is the number of times word t occurs in document d_i and df_t is the number of documents in the collection in which the word t occurs. The weight is then normalized by a document length. This model is often called the “bag-of-words model” because the factorial expression reflects conditional independence assumptions about word occurrences in d_i.

An identified cluster is represented in the similar way of that of document. A cluster is represented by a mean vector μ_j, often called a centroid or a cluster center, of documents which are grouped by their close similarity. We assume that there are K-clusters $C = \{\vec{\mu}_0, ..., \vec{\mu}_K\}$, which optimally partition a given data collection. Thus the goal of document clustering is to partition N number of documents into K clusters disjointly.

When text document and cluster are represented by the vector space model, the similarity of those vectors is determined by estimating the cosine angle between two vectors.

$$\cos(\vec{d}_i, \vec{\mu}_j) = \frac{\vec{d}_i \cdot \vec{\mu}_j}{|\vec{d}_i||\vec{\mu}_j|} = \frac{\sum^T d_{i,t} \times \mu_{j,t}}{\sqrt{\sum^T (d_{i,t})^2} \sqrt{\sum^T (\mu_{j,t})^2}} \tag{2}$$

3 Methods for Clustering

This section details each of the clustering methods that we used for text clustering. They are categorized into two classes: hierarchical and iterative clustering.

3.1 Hierarchical Clustering Methods

Hierarchical clustering proceeds either bottom-up, by starting with the individual instances and grouping the most similar ones, or top-down, by starting with all the objects and dividing them into groups so as to maximize the given objective function.

3.1.1 Hierarchical Agglomerative Clustering

Hierarchical Agglomerative Clustering (HAC) is a sequence of partitions in which each partition is nested into the next partition in the sequence [8]. Agglomerative clustering is defined by disjointed clustering, which individualizes each of the N documents within a cluster. This process is repeated to form a sequence of nested clusters in which the number of clusters decreases as the sequence progresses, until a single cluster containing all N documents. It consists of three combining methods, differentiated in terms of how they group data at each sequence of clustering; single-link groups the closest members, complete-link groups the farthest members, and average-link groups the closest members on average. The resulting tree of clusters is called a dendrogram, which shows how the clusters are related to each other. By cutting the dendrogram at a desired level, a clustering of the data items into disjoint groups is obtained.

3.1.2 Principal Direction Divisive Partitioning

Principal Direction Divisive Partitioning (PDDP) is a hierarchical divisive clustering [4]. It constructs a binary tree in which each node holds the documents. PDDP binary tree starts with the root cluster representing all documents. The algorithm then recursively splits each leaf cluster into two children, until a given criterion is satisfied. In order to keep the binary tree balanced, PDDP uses the scatter function to determine whether the node is partitioned. The scatter function measures how cohesive the instances within a cluster are. For example, if the mean squared distance of a cluster is greater than a given threshold value, the cluster should be partitioned. The word-by-document matrix is used to obtain the principal direction and the hyper-plane partition that is used to split the documents within a given node into two partitions.

For instance, we have the original word-by-document ($M = T \times N$) at the beginning of partitioning. In order to split a matrix into two sub-matrices (or nodes), each document is projected on the leading principal direction. The principal directions of the matrix is the eigenvectors $e = \{\vec{e}_1, ..., \vec{e}_T\}$ of its covariance matrix $\Sigma = (\vec{d} - \vec{\mu})(\vec{d} - \vec{\mu})^T$. The projection of document $\vec{d}_i$ is applied as:

$$v = \vec{e}(\vec{d}_i - \vec{\mu})$$

where v is the value which is used to determine the splitting for the cluster and $\vec{\mu}$ is the centroid of the matrix. All the documents for which $v \leq 0$ are partitioned into the

left child node, and all the documents for which $v > 0$ are put into the right child. The projection can be interpreted by the fact that we can obtain the hyperplane, which divides a set of multi-dimensional vectors into two distinct groups by projecting the vector d_i onto the line in the direction of $\vec{e}$ that passes through the centroid.

3.2 Iterative Clustering Methods

Iterative clustering usually produces clusters by optimizing an objective function defined either locally (among documents belong the same cluster) or globally (defined over all of the data set) [3]. The most popular objective function is the mean squared distance function, which tends to work well with isolated and compact clusters. The basic algorithm of iterative clustering works as follows:

1. Choose k cluster centers by either the random initialization or pick up randomly a number of documents and calculate their center,
2. Assign each document to the closest cluster center,
3. Recompute the cluster centers using the current cluster members,
4. If an objective criterion is not met, go to step 2.

The typical stopping criteria of this iteration are that there are no re-assignment of instances to new clusters or the mean squared distance of a cluster is less than the predefined threshold. A major problem of the iterative clustering is that it is very sensitive to the initialization and consequently converges to a local minimum of the target function value if the initial clusters are not properly chosen.

3.2.1 K-Means

The k-means is a well-known iterative clustering algorithm based on iterative relocation that partitions a data set into k clusters, minimizing the mean squared distance between the instances and the cluster centers. Given a number of documents $D = \{\vec{d}_0, ..., \vec{d}_N\}, \vec{d}_i \in R^T$, the k-means algorithm creates k-partitioning so that if $\{\vec{\mu}_1, ..., \vec{\mu}_k\}$ represent the k partition centers, then the following objective function [3]

$$J = \arg\min_j \sum_{j=1}^{k} \sum_{\vec{d}_i \in D} ||\vec{d}_i - \vec{\mu}_j||^2$$

is (locally) minimized.

3.2.2 Expectation-Maximization with Mixture Model

The underlying assumption of the mixture model [2] is that a given document collection, $D = \{\vec{d}_1, \vec{d}_2, ..., \vec{d}_n\}$, is generated by a set of k unknown distributions $n_1, n_2, ..., n_k$, which are represented by their parameters,$\Theta = \{\theta_1, \theta_2, ..., \theta_k\}$. In particular, the probability density of $\vec{d}_i$ with respect to θ_j is given by $P_j(\vec{d}_i|\theta_j)$ for some unknown set

of parameters θ_j. In addition, suppose that the probability that $\vec{d_i}$ belongs to a distribution θ_j is $z_{i,j}$. Given these definitions, the goal of the mixture model is to find the parameters θ and $z_{i,j}$ that maximize the likelihood:

$$L(\theta, z) = \prod_{i=1}^{n} \sum_{j=1}^{k} z_{i,j} P_k(\vec{d_i}|\theta_j)$$

The parameters of θ_j are the mean vector (μ_j) and covariance (Σ_j) of each distribution if the unknown distribution is modeled by the multivariate normal distribution. It is often called the Gaussian Mixture model. In this model, the model parameters can be estimated by the Expectation Maximization (EM) algorithm [5].

The EM is a general method of finding the maximum-likelihood (ML) estimate of the parameters of an underlying distribution (i.e. for estimating Gaussian mixtures) from a set of observed data that has incomplete or missing values (e.g. unlabeled) [6], [11]. In the E-step, the probability of each observation belonging to each cluster is estimated conditionally on the current parameter estimates. In the M-step, the model parameters are estimated given the current group membership probabilities. When the EM converges, each observation is assigned to the cluster with the maximum conditional probability.

In the clustering text documents, we assume that there is an unknown number of Gaussian distributions governing the generation of each document in a given data set. Each Gaussian distribution corresponds with a cluster. The goal of the EM process is to find the parameters, mean vector ($\vec{\mu_i}$), covariance matrix (Σ_i), and probability of each document belonging to a Gaussian distribution ($z_{i,j}$).

3.2.3 Constructive-Competition Clustering

The proposed method for clustering, called Constructive-Competition Clustering (C3), is motivated by constructive and competitive learning from neural network research [7]. When a multi-layered neural network is trained under the “constructive” fashion, a hidden node is dynamically appended to network architecture when it detects the intrinsic difference between input patterns and the learned weights. Correlation analysis is often used to discover intrinsic difference. The proposed method is also closely related to competitive learning in that the adjustment of cluster centers is confined to the single cluster center most similar to the instance presented.

Informed by these analogies, constructive-competition clustering first identifies the desired number of cluster centers by adding the new cluster center incrementally as long as there is a substantial dissimilarity between the existing cluster centers and the presented instance. We made use of cosine-measure (equation 2) and a empirical threshold to detect dissimilarity. In other words, if the cosine angle between a document and the center of cluster discovered is greater than threshold δ, then two vectors are considered to be dissimilar. The assignment of documents’ membership is deferred until it finishes the “construction” of the desired number of cluster centers. Each of the k desired number of cluster centers is discovered and normalized $||\vec{\mu_j}|| = 1, (j = 1, ..., k)$. When a new instance is presented, each of the clusters calculates its similarity by using equation 2. The most similar cluster is permitted to update

Input:
- Desired number of clusters, dc,
- Learning rate, η and its initial value, η_0
- Current number of clusters, cc,
- Maximum iterations = T, Current iteration = t
- Threshold, δ

Construction:

$\vec{\mu}_j \leftarrow \vec{d}_1$

Do while $cc <> dc$

1. $J = \arg\max_j cos(\vec{d}_i, \vec{\mu}_j)$
2.
    - Create a new cluster, if substantially dissimilar, $J > \delta, \vec{\mu} \leftarrow \vec{d}_i, ||\vec{\mu_j}|| = 1$
    - Update the weights of j cluster center, otherwise $\vec{\mu}_{t,j} \leftarrow \vec{\mu}_{t,j} + \eta\vec{d}_{t,i}$
3. Go to Step 1.

Competition:

$t \leftarrow 0$

Do while $\eta <> 0$

$\eta(t) \leftarrow \eta_0(1 - \frac{t}{T})$

1. Assign $\vec{d}_i$ to $\vec{\mu}_J$, $J = \arg\max cos(\vec{d}_i, \vec{\mu}_j)$
2. Update weight, $\vec{w}_j \leftarrow \vec{w}_j + \eta(t)\vec{d}_i$
3. Normalize weight, $\vec{w}_j \leftarrow \vec{w}_j/||\vec{w}_j||$

$t \leftarrow t + 1$

Figure 1: Constructive-Competition Clustering Algorithm.

its weights. The winner's weights are updated as:

$$\vec{w}_j \leftarrow \vec{w}_j + \eta(t)\vec{d}_i$$

where η is a learning rate at the iteration t. The weights are then normalized to ensure $\sum_{i=1}^{|T|} w_i^2 = 1$. Throughout this "competition" among the cluster centers, each cluster center moves toward the optimal cluster center in equivalence with the learning rate. In summary, the procedure of the constructive-competition clustering consists of two phases: "construction" and "competition" phases. In the constructive phase, it tries to find the desired number of cluster centers by adding a new cluster center whenever a substantial dissimilarity is discovered. Then it makes the discovered cluster centers move toward the optimal cluster centers by putting the cluster centers into the competition with one another. The "construction" phase is one of the solutions for the

fluctuated results that occurred by the random initialization. Figure 1 describes the algorithm in detail.

4 Experimentation

4.1 Data Sets

The data sets used in this paper are comprised of three real world data sets: one numerical data set and two textual data sets. The combination of data sets with different characteristics is intended to verify the applicability of the proposed method.

4.1.1 Heart Disease

The "Heart Disease" data set [1] has come from a study of heart disease screening and has been used for various classification and clustering tasks in the machine learning field. Each instance consists of 76 attributes with real value. The "label" is the first attribute in the instance: 0 = no heart disease, 1 to 4 = heart disease (different types or severities of the heart disease). Accordingly, there are five possible classes. The remaining attributes indicate features such as, history of disease in the family, smoking/non-smoking, and results of several medical tests. There are four separate data sets, all coming from the same study, but from different hospitals with very different demographics. We combined them in order to facilitate computation without problems since their format is identical. The total number of instances is 920.

4.1.2 20 Newsgroups

The 20 Newsgroups data set [2] is a collection of approximately 20,000 newsgroup articles, partitioned (nearly) evenly across 20 different newsgroups [9]. Except for a small fraction of the articles, each document belongs to exactly one newsgroup.

4.1.3 TDT Pilot Corpus

The Topic Detection Tracking (TDT) Pilot Study Corpus [3] comprises a set of stories that includes both newswire (text) and broadcast news (speech). Each story is represented as a stream of text, in which the text is either taken directly from Reuters or is a manual transcription of CNN. It consists of 15,863 chronologically-ordered news stories spanning the period from July 1, 1994 to June 30, 1995. There are 25 events manually identified in this corpus. Each story was assigned a label of "YES", "NO" or "BRIEF" with respect to each of the 25 events.

output \ target	Yes	No
Yes	a	b
No	c	d

Table 1: A contingency table for evaluating a binary classification. In the table, the column a is the number of items that match a (i.e. clustering) method's outputs and target values (i.e. the number of outputs correctly classified or clustered), b and c are the number of items that mismatch a method's outputs and target values respectively.

4.2 Evaluation Measures

The performance of the clustering methods is evaluated statistically in terms of how well the documents belonging to each of the target clusters match the documents belonging to the corresponding cluster. This presents a problem, because it is not known which of the clusters corresponds to a particular target cluster. Therefore it is required to associate each true label with (exactly) each of cluster outputs to determine this correspondence. This was accomplished by associating each target label with the cluster that best matches it. The degree of match between a target label and a cluster is defined as the number of instances that belong to both the target label and the cluster [1].

After finishing the association of the clustering result with corresponding target values, the contingency table 1 for a cluster is then used to measure a clustering result by means of the standard metric from Information Retrieval [10]. In order to evaluate the global performance of a clustering algorithm, we first sum up the scores in each cell for all clusters to make a single contingency table and then the following five metrics are computed for the single table.

- Recall, $r = \frac{a}{a+c}$,
- Precision, $p = \frac{a}{a+b}$,
- F1, $F1 = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$,
- Miss, $m = \frac{c}{a+c}$, and
- False Alarm, $f = \frac{b}{b+d}$

4.3 Experimental Results

The first experiment was carried out by using the non-text data set: "Heart Disease." Each instance consists of a number of (real-valued) attributes and is represented by matrix-form, $attributes \times instances$. We see the characteristic of this data set is very similar to that of textual data sets, even though the degree of dimensionality and the relationship among the attributes are considered to be different.

[1]The Heart Disease data set is publicly available at http://www.ics.uci.edu/∼mlearn/MLRepository.html.
[2]The 20 Newsgroups data set is available at http://www-2.cs.cmu.edu/∼TextLearning/datasets.html
[3]The TDT pilot corpus is available via the Linguistic Data Consortium (http://www.ldc.upenn.edu/)

	1	2	3	4	5	6	7	8	9	10
HAC	.8	.8	.8	.8	.8	.8	.8	.8	.8	.8
k-means	.64	.51	.62	.43	.54	**.72**	.61	.63	.54	.51
EM	.65	.61	.54	**.76**	.74	.62	.73	.81	.75	.61

Table 2: The micro-averaged accuracy results of three clustering methods were presented after 10 different runs on the heart disease data set. In this experiment, the optimal number of clusters which partitions the "Heart Disease" data set into subsets disjointly is 5. The average-link of hierarchical clustering is used.

	r	p	a
HAC	.54	.62	.80
k-means	.40	.41	.72
EM	.62	.64	.76
HAC with k-means	.65	.69	.79
HAC with EM	**.72**	**.82**	.81
PDDP	.63	.75	.83
C3	.65	.78	**.84**

Table 3: Clustering Results from the Heart-Disease data set. The threshold of determining the creation of a new cluster in the constructive-competition clustering was 0.8.

As mentioned earlier, the main drawback in using the iterative clustering methods is that its performance is very sensitive to initialization of the cluster center. Thus It is desirable to initially have more reliable center of a cluster. The clustering results by hierarchical methods might be a good source for the initial cluster centers of an iterative clustering. By staring up from the better-than-average ground, it is highly likely for an iterative clustering to find the global optimum.

Table 2 shows a result from comparison of three clustering methods after carrying out 10 different runs on the heart disease data set. The performance of iterative methods measured in micro-averaged accuracy are fluctuated between 50% and 80%, whereas those of hierarchical method showed a stabilized accuracy at 80%. Since we consider the iterative clustering procedure to be the optimization of the objective function, it is natural to think that it is heavily dependent on the initial search points to find the global optimum. This confirmed our assumptions that the iterative clustering methods, such as k-means and EM, are sensitive to the initialization of the cluster centers and such fluctuating results of iterative methods can be compensated for by combining the hierarchical and the iterative clustering method. Table 3 showed that the combined method showed a better performance than those of separate ones.

Table 4 and 5 show the clustering results on the text data sets: "20 Newsgroup" and "TDT pilot corpus," respectively. The results on both tables were generated by not allowing the duplicate association between a output and a target cluster. As we showed before, only the best matched pair of the output and the target was eligible for the evaluation. To construct the word feature set (W), the words are removed if they

	r	p	$F1$
HAC	.52	.54	.56
k-means	.49	.43	.55
EM	.50	.56	.60
HAC with k-means	.55	.57	.62
HAC with EM	.57	.53	.63
PDDP	.61	**.64**	.66
C3	**.62**	.62	**.69**

Table 4: Clustering Results from the 20 news group data set were presented. The optimal number of clusters in this experiment is 20.

	r	p	m	f	$F1$
HAC	.53	.56	.47	.20	.61
k-means	.50	.48	.50	.29	.57
EM	.52	.57	.48	.30	.62
HAC with k-means	.54	.59	.46	.10	.67
HAC with EM	.60	.66	.40	.30	.65
PDDP	.59	.70	.27	.11	.68
C3	**.69**	.74	**.32**	.09	**.72**
CMU	.62	**.82**	.38	**.04**	.71
UMass	.34	.53	.66	.09	.42
Dragon	.61	.69	.39	.08	.65

Table 5: The result was shown from clustering on the TDT pilot study corpus.

occur more than 300, *frequent threshold*, or less than 3 times, *infrequent threshold* from both text data sets. The text documents in both textual data sets are represented by the (real-valued) vector space model.

In the Newsgroup data experiment, only the body of each news article, which is the part of an article between the "Lines" field and the end of lines, was used for the experiment. PDDP and the proposed method showed better performance than the others. The TDT pilot corpus experiment showed similar tendencies. The result was shown from the TDT pilot study corpus and the results from other works on the last three rows are cited from [13], in order to compare performance. The desired number of clusters is 25, as indicated in [13], but there are more clusters in the TDT pilot corpus beyond the 25 labeled ones. This means that the clustering method may detect many potential events, but only that which was evaluated on a subset of the system-generated clusters best matched the manually-labeled events.

5 Conclusions

In this paper, we proposed a new clustering method and compared its performance with the existing methods. Generally a hierarchical clustering showed more stable results than an iterative one, but it takes more time to generate the result due to its quadratic time complexity. The computing time of an iterative clustering is linear in the number of documents. Through the experiments, we confirmed that the performance of iterative clustering methods fluctuated depending on the initialization of cluster centers and it is possible to improve its performance by starting the clustering from the result of the hierarchical clustering. The combined method showed better performance than that of each method alone, but the performance of the experiments on the text data sets is not promising.

To compensate for these drawbacks of the two major clustering methods, we proposed the constructive-competition clustering algorithm. It is motivated by constructive and competitive learning from the neural network research. For the construction phase, it tries to find the desired number of clusters by adding a new cluster when there is a intrinsic difference between the instance presented and the existing clusters. Each cluster centers then moves toward the optimal cluster center in equivalence with the learning rate by adjusting its weight vector. From the experimental results on the three different real world data sets, the proposed method shows an even trend of performance across the different domains, while the performance of our algorithm on text domains was better than that reported in previous research.

Acknowledgments

This work has been partially supported by DARPA grant F30602-98-2-0138 and by AFOSR grant F49620-01-1-0542.

References

[1] J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, 1998.

[2] J. Banfield and A. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, 49:803–821, 1993.

[3] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.

[4] D. Boley. Principal direction divisive partitioning. *Data Mining and Knowledge Discovery*, 2:325–344, 1998.

[5] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society*, 39:1–38, 1977.

[6] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. John Wiley and Sons, 2001.

[7] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall, 1999.

[8] A. Jain and R. Dubes. *Algorithms for Clustering Data*. Prentice Hall, 1988.

[9] K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of International Conference on Machine Learning (ICML-95)*, pages 331–339, 1995.

[10] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[11] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[12] G. Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison-Wesley, 1989.

[13] Y. Yang, J. Carbonell, R. Brown, T. Pierce, B. Archibald, and X. Liu. Learning approaches for detecting and tracking news events. *IEEE Intelligent Systems: Special Issue on Application of Intelligent Information Retrieval*, 14(4):32–43, 1999.